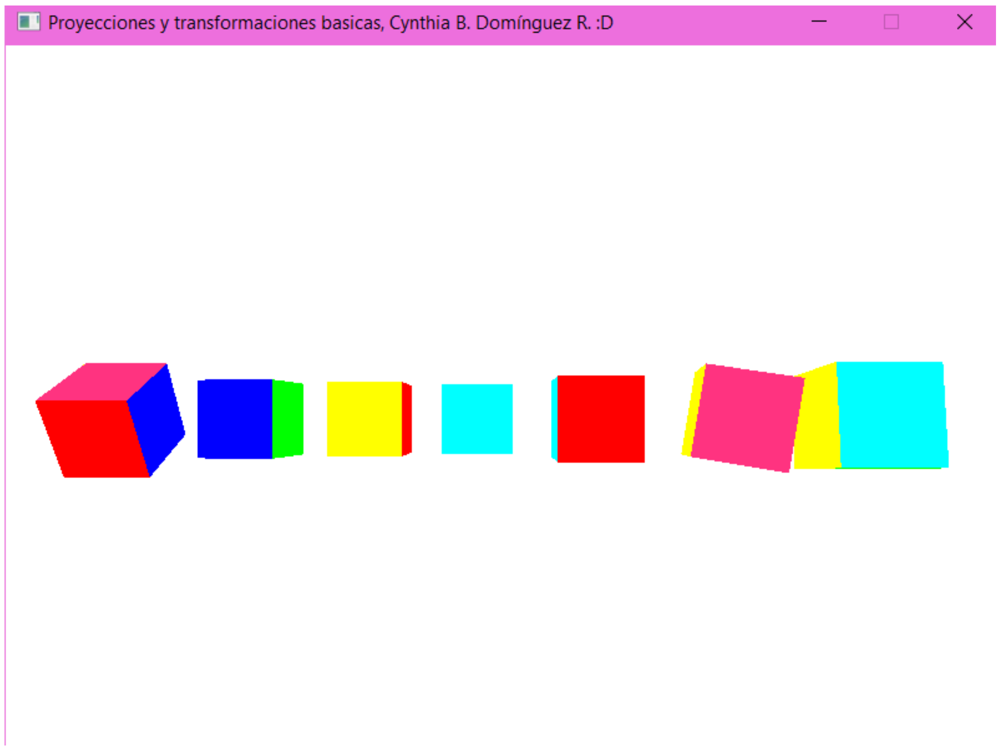
Práctica2 Domínguez Reyes Cynthia Berenice
22 – agosto – 24 314148632

Transformaciones

Desarrollo:

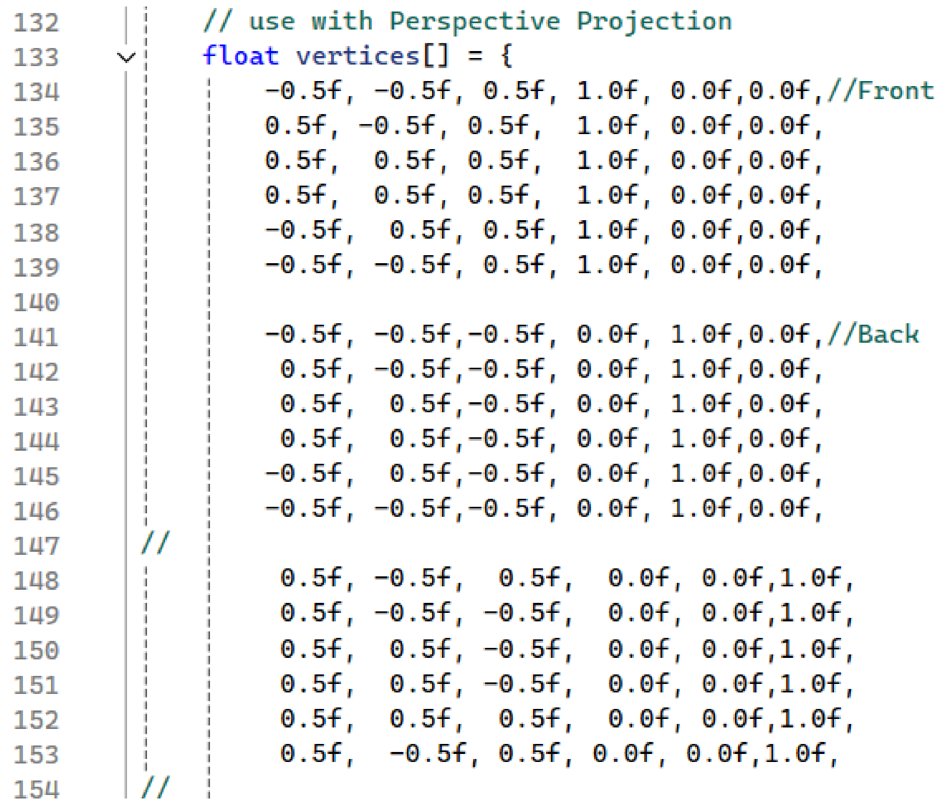
Para esta práctica, trabajamos con las vistas ortogonales y con las vistas en perspectiva. Para el ejercicio en clase, generamos cubos en perspectiva que simularan la siguiente imagen:

Dependiendo las letras de nuestro nombre serían la cantidad de cubos que generaríamos, en mi caso son siete cubos por C Y N T H I A, los cuales podemos observarlos en la siguiente imagen:

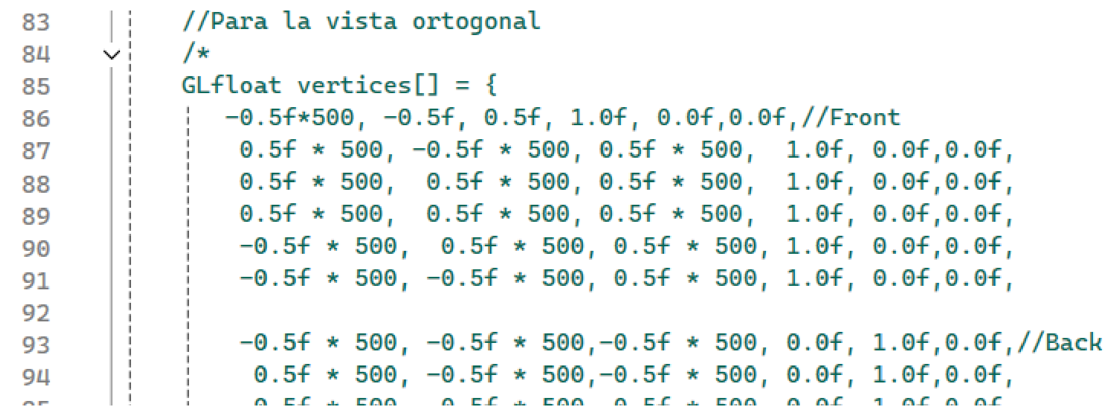

Proyecciones y transformaciones basicas, Cynthia B. Domínguez R. :D

Ajustes realizados

En el código teníamos dos arreglos, uno para la vista ortogonal y otro para la vista en perspectiva, el de la vista ortogonal se comenta para que no afecte a la ejecución del código y utilizamos el de la vista en perspectiva.

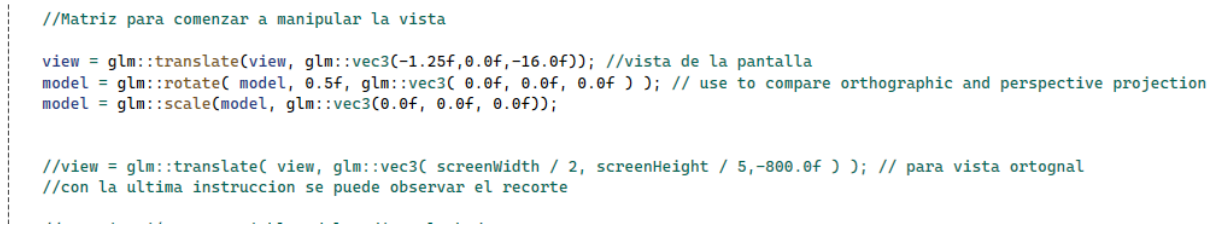


```
    // use with Perspective Projection
    float vertices[] = {
        -0.5f, -0.5f, 0.5f, 1.0f, 0.0f,0.0f,//Front
        0.5f, -0.5f, 0.5f,  1.0f, 0.0f,0.0f,
        0.5f,  0.5f, 0.5f,  1.0f, 0.0f,0.0f,
        0.5f,  0.5f, 0.5f,  1.0f, 0.0f,0.0f,
        -0.5f,  0.5f, 0.5f, 1.0f, 0.0f,0.0f,
        -0.5f, -0.5f, 0.5f, 1.0f, 0.0f,0.0f,

        -0.5f, -0.5f,-0.5f, 0.0f, 1.0f,0.0f,//Back
         0.5f, -0.5f,-0.5f, 0.0f, 1.0f,0.0f,
         0.5f,  0.5f,-0.5f, 0.0f, 1.0f,0.0f,
         0.5f,  0.5f,-0.5f, 0.0f, 1.0f,0.0f,
        -0.5f,  0.5f,-0.5f, 0.0f, 1.0f,0.0f,
        -0.5f, -0.5f,-0.5f, 0.0f, 1.0f,0.0f,
//
         0.5f, -0.5f,  0.5f,  0.0f, 0.0f,1.0f,
         0.5f, -0.5f, -0.5f,  0.0f, 0.0f,1.0f,
         0.5f,  0.5f, -0.5f,  0.0f, 0.0f,1.0f,
         0.5f,  0.5f, -0.5f,  0.0f, 0.0f,1.0f,
         0.5f,  0.5f,  0.5f,  0.0f, 0.0f,1.0f,
         0.5f,  -0.5f, 0.5f, 0.0f, 0.0f,1.0f,
//
```



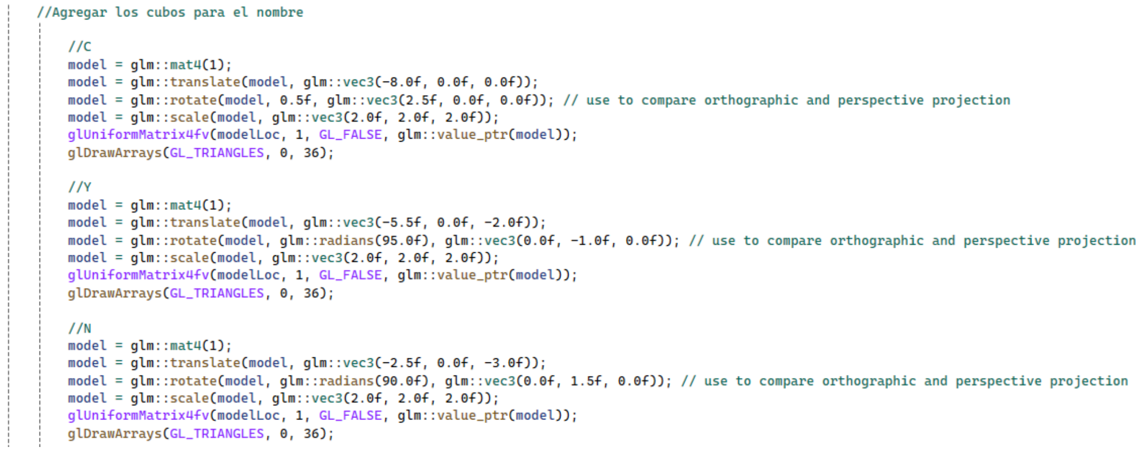
```
    //Para la vista ortogonal
    /*
    GLfloat vertices[] = {
        -0.5f*500, -0.5f, 0.5f, 1.0f, 0.0f,0.0f,//Front
         0.5f * 500, -0.5f * 500, 0.5f * 500,  1.0f, 0.0f,0.0f,
         0.5f * 500,  0.5f * 500, 0.5f * 500,  1.0f, 0.0f,0.0f,
         0.5f * 500,  0.5f * 500, 0.5f * 500,  1.0f, 0.0f,0.0f,
        -0.5f * 500,  0.5f * 500, 0.5f * 500, 1.0f, 0.0f,0.0f,
        -0.5f * 500, -0.5f * 500, 0.5f * 500, 1.0f, 0.0f,0.0f,

        -0.5f * 500, -0.5f * 500,-0.5f * 500, 0.0f, 1.0f,0.0f,//Back
         0.5f * 500, -0.5f * 500,-0.5f * 500, 0.0f, 1.0f,0.0f,
         0.5f * 500,  0.5f * 500, 0.5f * 500, 0.0f, 1.0f,0.0f,
```

Para las funciones también es necesario comentar la parte de la vista ortogonal y dejar funcionando las de la vista perspectiva.

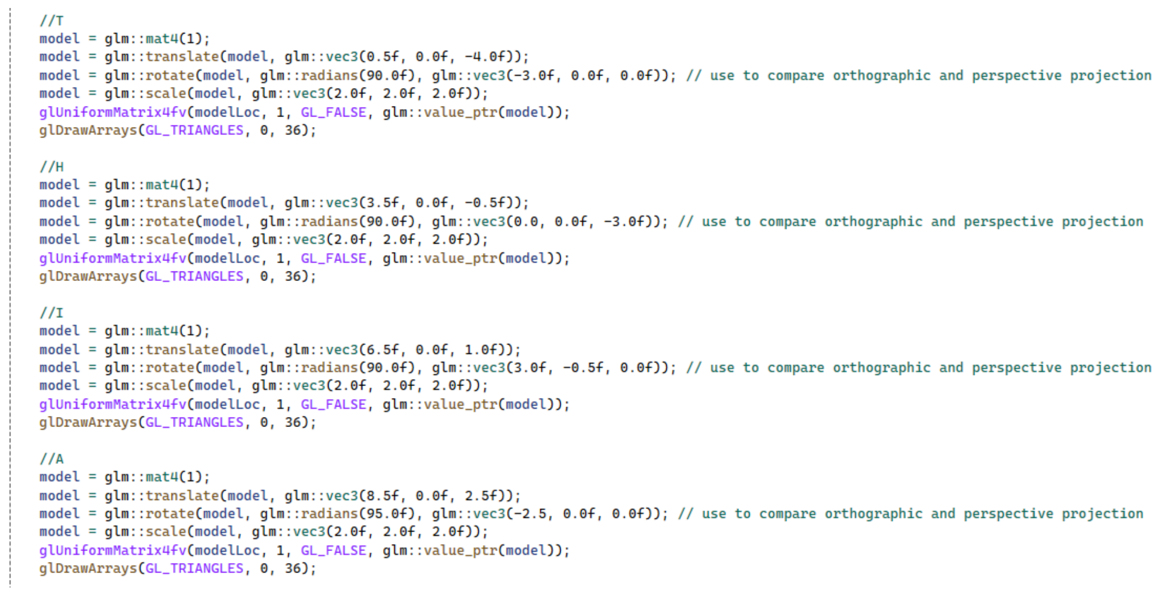


```
//Matriz para comenzar a manipular la vista

view = glm::translate(view, glm::vec3(-1.25f,0.0f,-16.0f)); //vista de la pantalla
model = glm::rotate( model, 0.5f, glm::vec3( 0.0f, 0.0f, 0.0f ) ); // use to compare orthographic and perspective projection
model = glm::scale(model, glm::vec3(0.0f, 0.0f, 0.0f));


//view = glm::translate( view, glm::vec3( screenWidth / 2, screenHeight / 5,-800.0f ) ); // para vista ortognal
//con la ultima instruccion se puede observar el recorte
```

En el código para cada cubo y saber cuál estaba generando, coloque la letra de mi nombre con respecto al cubo que estaba dibujando, de la siguiente manera:

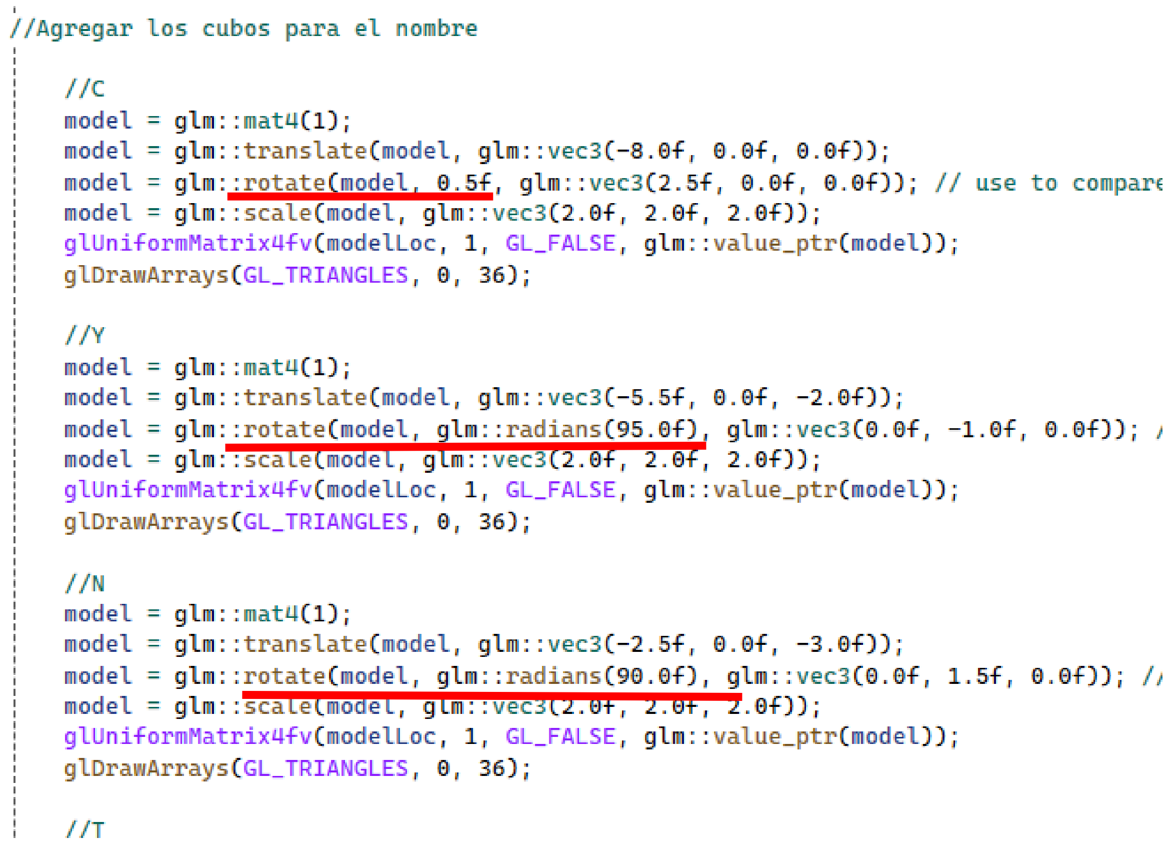


```
//Agregar los cubos para el nombre

    //C
    model = glm::mat4(1);
    model = glm::translate(model, glm::vec3(-8.0f, 0.0f, 0.0f));
    model = glm::rotate(model, 0.5f, glm::vec3(2.5f, 0.0f, 0.0f)); // use to compare orthographic and perspective projection
    model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
    glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
    glDrawArrays(GL_TRIANGLES, 0, 36);

    //Y
    model = glm::mat4(1);
    model = glm::translate(model, glm::vec3(-5.5f, 0.0f, -2.0f));
    model = glm::rotate(model, glm::radians(95.0f), glm::vec3(0.0f, -1.0f, 0.0f)); // use to compare orthographic and perspective projection
    model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
    glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
    glDrawArrays(GL_TRIANGLES, 0, 36);

    //N
    model = glm::mat4(1);
    model = glm::translate(model, glm::vec3(-2.5f, 0.0f, -3.0f));
    model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.5f, 0.0f)); // use to compare orthographic and perspective projection
    model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
    glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
    glDrawArrays(GL_TRIANGLES, 0, 36);
```



```
//T
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(0.5f, 0.0f, -4.0f));
model = glm::rotate(model, glm::radians(90.0f), glm::vec3(-3.0f, 0.0f, 0.0f)); // use to compare orthographic and perspective projection
model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//H
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(3.5f, 0.0f, -0.5f));
model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0, 0.0f, -3.0f)); // use to compare orthographic and perspective projection
model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//I
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(6.5f, 0.0f, 1.0f));
model = glm::rotate(model, glm::radians(90.0f), glm::vec3(3.0f, -0.5f, 0.0f)); // use to compare orthographic and perspective projection
model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);

//A
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(8.5f, 0.0f, 2.5f));
model = glm::rotate(model, glm::radians(95.0f), glm::vec3(-2.5, 0.0f, 0.0f)); // use to compare orthographic and perspective projection
model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glDrawArrays(GL_TRIANGLES, 0, 36);
```

En cada bloque del código para dibujar un cubo, modifique la función rotate para hacer que el cubo girara en grados a través de la función radians, para todos los cubos utilice 90°, excepto el primero y el segundo, que deje la configuración como la teníamos y en el segundo coloque 95°.



```
//Agregar los cubos para el nombre

    //C
    model = glm::mat4(1);
    model = glm::translate(model, glm::vec3(-8.0f, 0.0f, 0.0f));
    model = glm::rotate(model, 0.5f, glm::vec3(2.5f, 0.0f, 0.0f)); // use to compare
    model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
    glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
    glDrawArrays(GL_TRIANGLES, 0, 36);

    //Y
    model = glm::mat4(1);
    model = glm::translate(model, glm::vec3(-5.5f, 0.0f, -2.0f));
    model = glm::rotate(model, glm::radians(95.0f), glm::vec3(0.0f, -1.0f, 0.0f)); /
    model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
    glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
    glDrawArrays(GL_TRIANGLES, 0, 36);

    //N
    model = glm::mat4(1);
    model = glm::translate(model, glm::vec3(-2.5f, 0.0f, -3.0f));
    model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.5f, 0.0f)); //
    model = glm::scale(model, glm::vec3(2.0f, 2.0f, 2.0f));
    glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
    glDrawArrays(GL_TRIANGLES, 0, 36);

    //T
```

Liga del git:

https://github.com/bereenicee7/Curso_Computacion_Grafica/blob/main/ConfigInicial/P2_Transformaciones.cpp